WHAT IS CLAIMED IS:

1. A method for providing dynamic symbolic link resolution, comprising:
receiving a first pathname;
determining if the first pathname is a dynamic symbolic link (DSL);
determining at least a first value associated with the DSL; and
substituting the first value into the first pathname producing a first target pathname.

2. The method as claimed in claim 1, further comprising the step of:
accessing a file designated by the first target pathname.

3. The method as claimed in claim 1, further comprising the step of:
extracting at least one tag from the DSL; and
utilizing the at least one tag in the step of determining at least the first value associated with the DSL.

4. The method as claimed in claim 3, wherein:
the step of extracting the at least one tag including searching the DSL for at least one DSL declaration.

5. The method as claimed in claim 3, wherein:
the step of extracting the at least one tag including searching the DSL for at least one predefined alphanumeric character sequence.

6. The method as claimed in claim 1, further comprising the step of:
resolving the first pathname as a symbolic link to the DSL prior to the step of determining if the first pathname is the DSL.

7. The method as claimed in claim 1, further comprising the step of:
returning a first file handle to the first target path.

8. The method as claimed in claim 1, wherein:
the step of receiving the first pathname including receiving the first pathname from a first application attempting to access a first file;
receiving the first pathname from a second application attempting to open the first file; and
returning a second file handle to a second target pathname.

9. The method as claimed in claim 8, further comprising the step of:
determining if the first pathname received from the second application is a DSL;
determining at least a second value associated with the DSL; and
substituting at least the second value into the first pathname producing the second target pathname following the step of receiving the first pathname from the second application.

10. The method as claimed in claim 9, wherein:
the first application is a first instance of the first application; and
the second application is a second instance of the first application.

11. The method as claimed in claim 1, further comprising the steps of:
registering the DSL prior to the step of receiving the first pathname including:
a) registering a DSL specification; and
b) recording the DSL specification.

12. The method as claimed in claim 11, further comprising the steps of:
receiving the DSL specification from a first process prior to the step of registering the DSL specification.

13. The method as claimed in claim 12, further comprising the steps of:
the first application inheriting the DSL specification of the first process.

14. The method as claimed in claim 1, further comprising the steps of:
generating the DSL prior to the step of receiving the first pathname including:
a) obtaining the first pathname;
b) renaming the first pathname as the first target pathname;
c) determining at least one variable in the first target pathname; and
d) creating a symbolic link from the first pathname pointing to the DSL.

15. The method as claimed in claim 14, further comprising the steps of:
substituting the at least one variable in the first target pathname with a DSL declaration.

16. The method as claimed in claim 1, further comprising the steps of:
determining a second value associated with the DSL; and

substituting the second value into the first pathname prior to producing the first target pathname.

17. The method as claimed in claim 1, further comprising the steps of:

determining in a default value if the first value can not be determined; and

substituting the default value into the first pathname producing the first target pathname.

18. The method as claimed in claim 1, wherein:

the step of determining if the first pathname is a DSL including determining if the pathname includes at least one declaration;

the step of determining the first value including:

a) extracting a tag from the at least one declaration; and

b) utilizing the tag as the first value; and

the step of substituting the first value into the first pathname including substituting the tag as the first value into the pathname producing the first target pathname.

19. The method as claimed in claim 1, wherein:

the step of determining if the first pathname is a DSL including determining if the pathname includes at least one declaration;

the step of determining the first value including utilizing the declaration as the first value; and

the step of substituting the first value into the first pathname including substituting the declaration as the first value into the pathname producing the first target pathname.

20. A computer system providing a method for providing a dynamic symbolic link, comprising the steps of:

renaming a first pathname to a target pathname;

determining a variable within the target pathname;

defining the first pathname as a symbolic link; and

associating the symbolic link with a virtual pathname.

21. The computer system as claimed in claim 20, further comprising the step of:

defining a specification associated with the virtual pathname.

22. The computer system as claimed in claim 20, wherein:
the step of defining a specification including associating the variable with the virtual pathname.

23. The computer system as claimed in claim 20, wherein:
the step of associating the symbolic link including defining a declaration within the virtual pathname.

24. The computer system as claimed in claim 20, further comprising the steps of:
resolving the dynamic symbolic link including:
a) receiving a request for the symbolic link,
b) determining the DSL;
c) determining the target pathname from the DSL; and
d) returning a handle to the target pathname.

25. The computer system as claimed in claim 24, wherein:
the step of determining the target pathname including:
a) extracting a DSL declaration from the DSL;
b) extracting a tag from the DSL declaration;
c) determining a value associated with the tag; and
d) substituting the value into the DSL to generate the target pathname.

26. A method for enabling a dynamic pathname, comprising:
receiving a request to access information stored under a pathname;
determining if the pathname is dynamic;
resolving the dynamic pathname; and
providing access to the information stored under the resolved pathname.

27. The method as claimed in claim 26, wherein:
the step of determining if the pathname is dynamic including determining if the pathname includes at least one declaration.

28. The method as claimed in claim 27, wherein:
the step of resolving the dynamic pathname including:
a) extracting a tag from the at least one declaration; and
b) determining a value associated with the tag.

29. The method as claimed in claim 28, wherein:
the step of providing access to the information including:
a) substituting the value for the at least one declaration; and
b) generating the resolved pathname.

30. The method as claimed in claim 27, wherein:
the step of determining if the pathname is dynamic including determining if the pathname includes a plurality of declarations.

31. The method as claimed in claim 26, wherein:
the step of resolving the dynamic pathname including:
a) extracting a tag for each of the plurality of declarations;
b) determining a value associated with each tag; and
the step of providing access to the information including:
a) substituting the associated value for each declaration; and
b) generating the resolved pathname.

32. The method as claimed in claim 26, further comprising the steps of:
creating the dynamic pathname including:
a) determining a target pathname;
b) determining at least one component of the target pathname to be dynamic;
c) generating the dynamic pathname including replacing the at least one component with a declaration; and
d) generating a symbolic link pointing to the dynamic pathname.

33. The method as claimed in claim 26, further comprising the steps of:
defining a specification associated with the at least one component including:
a) storing a tag; and
b) storing a value associated with the tag.

34. A computer program product for providing access to stored information, the computer program product including a computer readable storage medium and a computer program mechanism embedded therein, the computer program mechanism comprising:
a method of providing dynamic symbolic links (DSL) comprising:
a) receiving a request to access information stored under a pathname;
b) determining if the pathname is dynamic;
c) resolving the dynamic pathname resulting in resolved pathname; and

d) providing access to the information stored under the resolved pathname.

35. The computer program product as claimed in claim 34, further comprising:
the step of determining if the pathname is dynamic including
a) determining if the pathname includes at least one declaration.

36. The computer program product as claimed in claim 35, further comprising:
the step of resolving the dynamic pathname including:
a) determining a first value associated with the at least one declaration; and
b) substituting the first value into the first pathname producing the resolved pathname.

37. The computer program product as claimed in claim 36, further comprising:
the step of resolving the dynamic pathname including:
a) determining a default value if the first value is not determined; and
b) substituting the default value into the first pathname producing the resolved pathname.

38. The computer program product as claimed in claim 36, further comprising:
the step of resolving the dynamic pathname including determining a tag from the declaration; and
determining the first value associated with the tag.

39. The computer program product as claimed in claim 34, wherein:
the step of providing access to the information including returning a file handle to the resolved pathname.

40. The computer program product as claimed in claim 34, further comprising:
generating the DSL prior to the step of receiving the request to access the information stored under the pathname including:
a) obtaining the pathname;
b) renaming the pathname as the resolved pathname;
c) determining at least one variable in the resolved pathname;
d) substituting the at least one variable in the resolved pathname with a declaration; and
e) creating a symbolic link from the pathname pointing to the DSL.

41. The computer program product as claimed in claim 36, further comprising:

registering the DSL prior to the step of receiving the request to access the information stored under the pathname including:

a) receiving a DSL specification from an application;

b) registering the DSL specification associated with the application; and

c) recording the DSL specification.

42. The computer program product as claimed in claim 41, wherein:

the step of receiving the DSL including receiving a tag and an associated value.

43. A computer network configured to provide dynamic symbolic link (DSL) and DSL resolution, the computational system comprising:

a processor configured to run at least an operating system and at least one application;

a memory coupled with the processor and configured to store information such that the information is stored and accessed through a pathname; and

a means for accessing the stored information through the pathname including a means for dynamically resolving the pathname.

44. The computer network as claimed in claim 43, further comprising:

the means for dynamically resolving the pathname including a means for determining if the pathname includes at least one declaration;

a means for extracting a tag from the at least one declaration;

a means for determining a value associated with the tag;

a means for substituting into the pathname the value for the declaration providing a target pathname; and

a means for accessing the target pathname including a means for returning a file handle of the target pathname to the application.